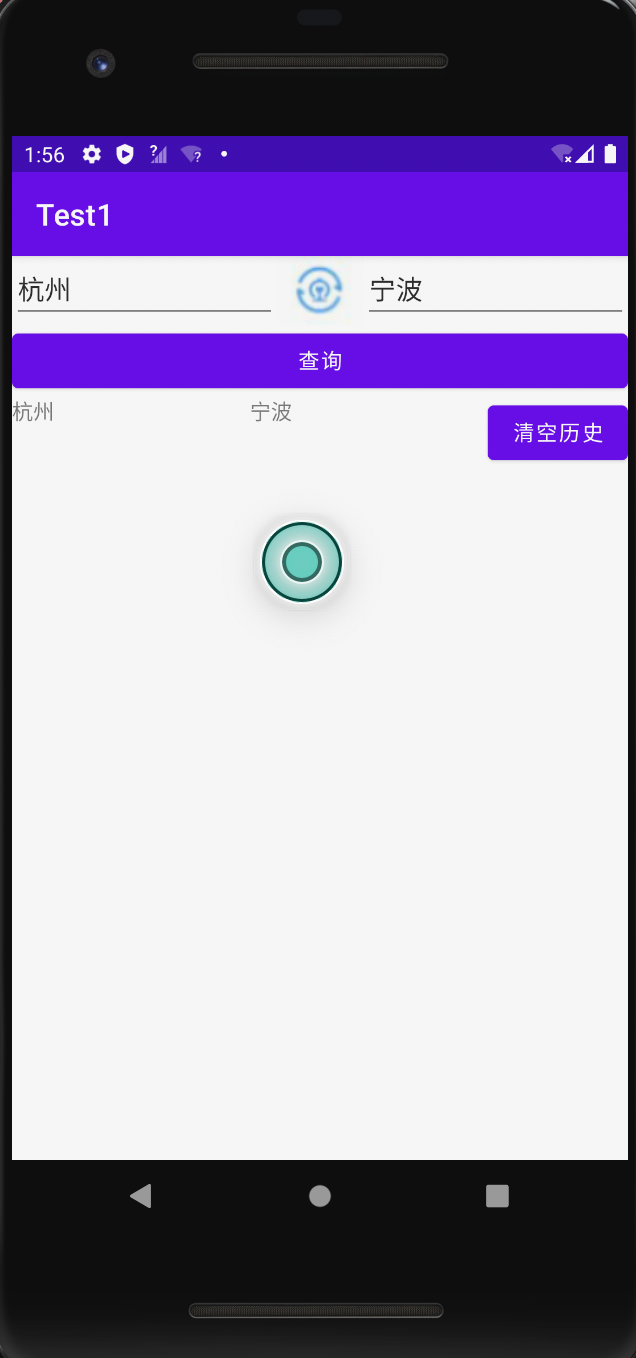
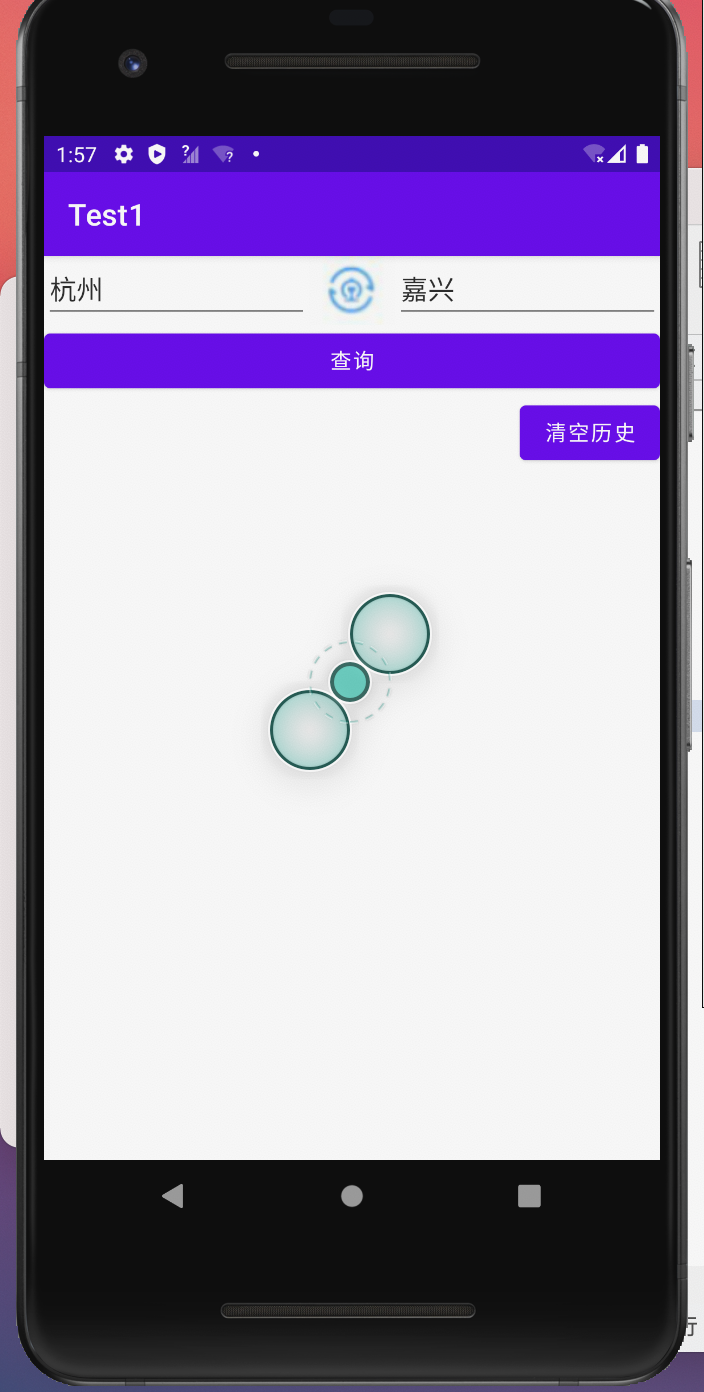
实验报告要求

1 实验内容

## 通过文件：实现查询历史记录的存储

2 实验效果





3 实验实现过程

1. 文件如何存储

fun save(binding: ActivityTrainSearchBinding){

try {

//打开文件

val output = openFileOutput(file\_name, Context.MODE\_APPEND)

// 缓冲流

val writer = BufferedWriter(OutputStreamWriter(output))

writer.use {

for(item in history\_data){

writer.write("${item.startStation},${item.endStation} ")//csv格式

writer.newLine()

}

}

Toast.makeText(this, "已写入", Toast.LENGTH\_LONG).show()

refreshHistory(binding)

}catch (e: IOException){

e.printStackTrace()

}

}

1. 文件如何读取

fun load(binding: ActivityTrainSearchBinding){

val builder = java.lang.StringBuilder()

try{

val input = openFileInput(file\_name)

val reader = BufferedReader(InputStreamReader(input))

reader.use {

reader.forEachLine {

builder.append(it).append(", ")

val info = parseHistory(it)

history\_data.add(info)

}

}

//binding.tvHistory.setText(builder.toString())

// val adapter = TrainAdapter(history\_data)

// binding.tvHistory.adapter = adapter

refreshHistory(binding)

}catch (e:IOException){

e.printStackTrace()

}

}

private fun parseHistory(history\_str:String):TrainSearchHistory{

val splitIndex = history\_str.indexOf(",")

val startStation = history\_str.substring(0,splitIndex).trim()

val endStation = history\_str.substring(splitIndex+1,history\_str.lastIndex).trim()

return TrainSearchHistory(startStation,endStation)

}

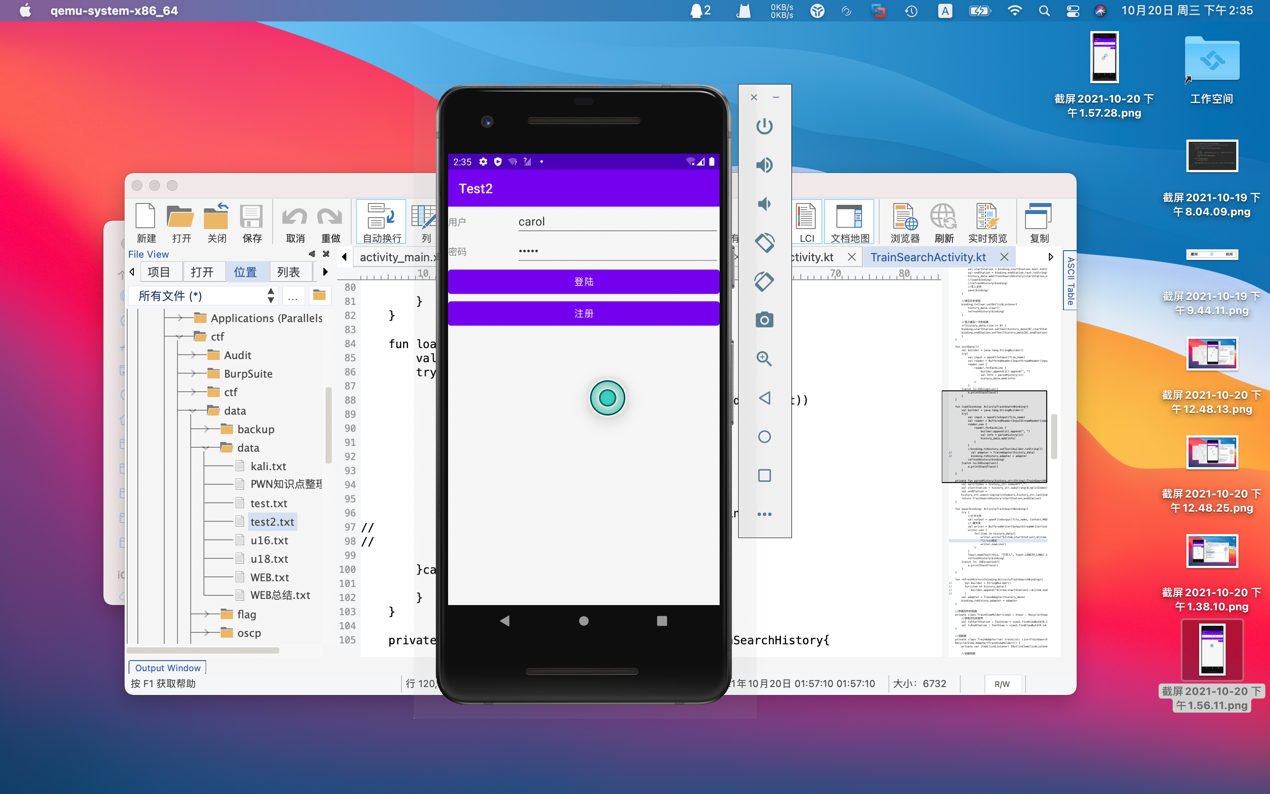
1. 查询历史记录保存和读取整体流程

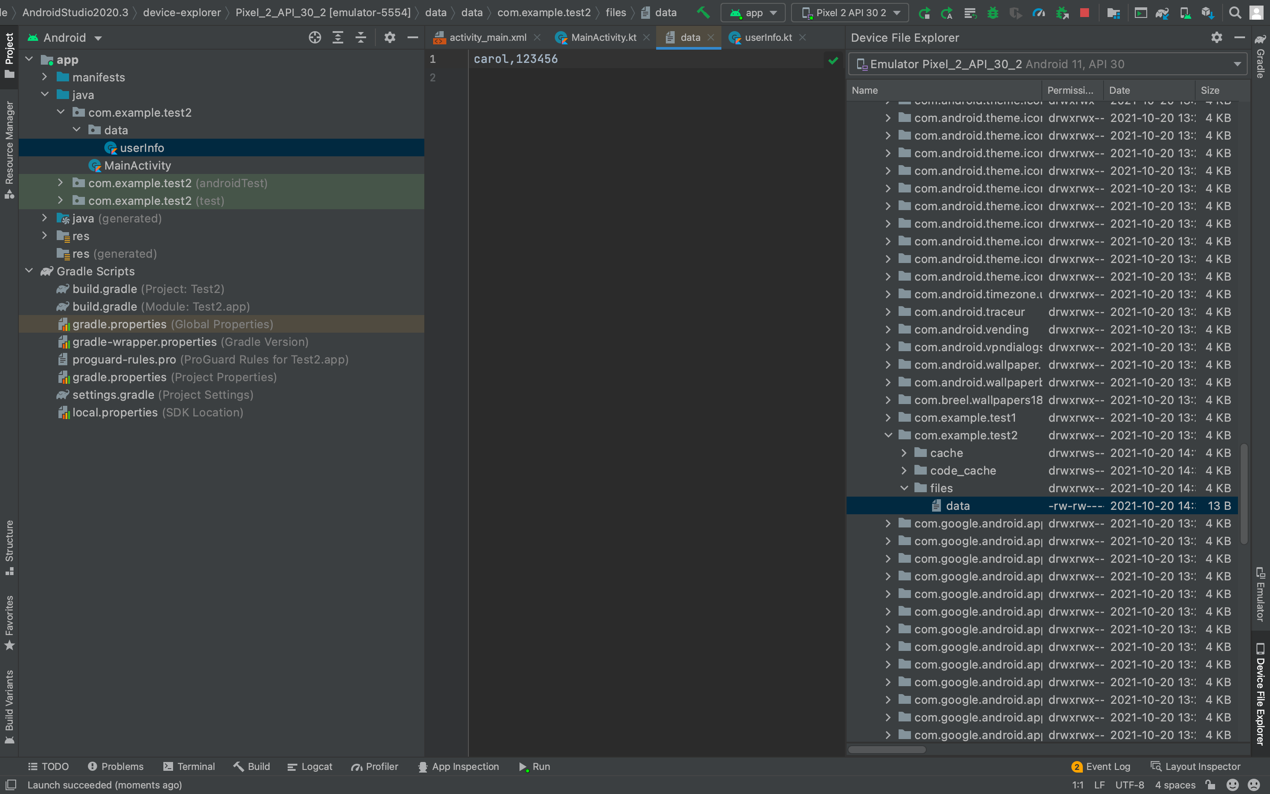
在构造函数里先initdata，然后绑定布局管理器和适配器，每次点击查询按钮，就把edittext里的内容写入文件，点击列表里对应的数据，就会填入edittext，点击清除历史按钮，就能清空历史记录。

1 实验内容

## 12306登陆界面的实现

2 实验效果





3 实验实现过程

1. 文件如何存储

private fun save(binding: ActivityMainBinding){

val username = binding.username.text.toString()

val password = binding.password.text.toString()

try {

//打开文件

val output = openFileOutput(file\_name, Context.MODE\_APPEND)

// //写入文件 二进制流

// output.write(data.toByteArray())

// 字符流

val outputStreamWriter = OutputStreamWriter(output)

val num = 100

// outputStreamWriter.use {

// it.write("数据：${data} -- ${num}")

// }

// 缓冲流

val writer = BufferedWriter(OutputStreamWriter(output))

writer.use {

it.write("${username},${password}")

it.newLine()

}

}catch (e: IOException){

e.printStackTrace()

}

Toast.makeText(this, "写入完毕", Toast.LENGTH\_LONG).show()

}

1. 文件如何读取

private fun read(binding: ActivityMainBinding){

val builder = StringBuilder()

var last:Int = 0

try{

val input = openFileInput(file\_name)

val reader = BufferedReader(InputStreamReader(input))

reader.use {

reader.forEachLine {

builder.append(it).append(", ")

val info = parseHistory(it)

userInfo\_data.add(info)

}

}

last = userInfo\_data.size-1

binding.username.setText(userInfo\_data[last].username)

binding.password.setText(userInfo\_data[last].password)

}catch (e:IOException){

e.printStackTrace()

}

}

1. 查询历史记录保存和读取整体流程

在构造函数的时候先进行一次文件读取，读取出用户名和密码，填入editText。当用户输入用户名和密码，点击登陆的时候，进行save。